НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ

«До захисту допущено»
В.о. завідувача кафедрою
_____ М.М.Савчук
(підпис) (ініціали, прізвище)
“___”_____20 _ р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки : 113 «Прикладна математика»
(код і назва)

на тему: Аналіз загроз при повторному використанні налаштування у протоколі GRO-16 _____
_____
_____

Виконав (-ла): студент (-ка) _4___ курсу, групи ____ФІ-62_____
(шифр групи)

Бещук Андрій Андрійович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент кафедри ммзи, професор, дтн, Ковальчук Л.В. _ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент доцент, к.ф.-м.н., доцент, Южакова А. А. _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020року

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»
Фізико-технічний інститут

Кафедра математичних методів захисту інформації

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки - 113 «Прикладна математика»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедрою

М.М.Савчук
_____
(підпис) (ініціали, прізвище)

«___»_____20__ р.

ЗАВДАННЯ
на дипломну роботу студенту

Бещуку Андрію Андрійовичу _____
(прізвище, ім'я, по батькові)

1. Тема роботи _____
_____
Аналіз загроз при повторному використанні налаштування у протоколі GRO-16,
керівник роботи Ковальчук Людмила Василівна, доцент технічних наук, професор _____ ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від _____ р. № _____

2. Термін подання студентом роботи 08.06.2020 _____

3. Вихідні дані до роботи _____
_____
_____
_____

4. Зміст роботи _____
_____
Робота складається з аналізу проблем способів забезпечення анонімності у найбільш поширених криптовалютах, детальним прикладом реалізації одного з таких способів у якості неінтерактивних стислих доведень без

розголошення на прикладі протоколу GRO-16, аналіз можливих схем етапу SETUP, аналіз їх вразливостей та розробка нових атак на протокол GRO-16.

_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) _____

_____

_____

_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Знайомство із засобами забезпечення анонімності у криптовалютах	10.09.2019-10.10.2019	
2.	Огляд літератури за темою анонімності у криптовалютах	10.10.2019-25.11.2019	
3.	Аналіз проблем забезпечення анонімності у найбільш поширених критовалютах	25.11.2019-10.02.2020	
4.	Практична реалізація протоколу GRO-16	10.02.2020-27.03.2020	
5.	Розробка атак на протокол GRO-16	27.03.2020-02.06.2020	

Студент _____ _____
(підпис) (ініціали, прізвище)

Керівник роботи _____ _____
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Кваліфікаційна робота містить: 63 стор., 1 рисунок, 0 таблиць, 15 джерел.

Метою даної роботи є аналіз вразливостей протоколу доведення без розголошення типу zk-SNARK GRO-16 та умов, за яких ці вразливості будуть критичними. Об'єктом дослідження є процес захисту персональних даних при виконанні транзакцій у блокчейні. Предметом дослідження є атаки на протокол захисту персональних даних у блокчейні за умови повторного використання налаштування.

Було покроково розглянуто етап SETUP доведень без розголошення типу zk-SNARK, та розроблені алгоритми побудови формування множини налаштування. Були проаналізовані вразливості сценарію розпаралелювання формування налаштування та розроблені п'ять атак на протокол GRO-16. Вони ґрунтуються на людському факторі та змові декількох учасників. Також, були запропоновані механізми захисту від наведених у цій роботі атак, як алгоритмічні, так і із використанням пристроїв третьої сторони.

ДОВЕДЕННЯ БЕЗ РОЗГОЛОШЕННЯ, ZK-SNARK, БЛОКЧЕЙН, АНОНІМНІСТЬ ТРАНЗАКЦІЙ У БЛОКЧЕЙНІ

ABSTRACT

Qualification work consists of: 63 pages, 1 picture, 0 tables, 15 sources.

Goal of this work is analysis of vulnerabilities zero knowledge proof protocols zk-SNARK type GRO-16 and conditions, in which those vulnerabilities are critical. The object of this research is process of protection personal data while performing transactions in the blockchain. The subject of this research is attacks on personal data protection protocol in blockchain subject to reuse setup.

The SETUP stage of zero knowledge proof protocols zk-SNARK type was considered step by step and setup constricting algorithms were developed. Vulnerabilities of generation setup parallelization were analyzed and five types of attack on protocol GRO-16 were developed. They are based on the human factor and the conspiracy of several participants. Also, strategies have been developed to protect against the attacks described in this work, both algorithmic and using third-party devices. ZERO KNOWLEDGE PROOF, ZK-SNARK, BLOCKCHAIN, ANONYMITY OF TRANSACTIONS IN BLOCKCHAIN

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ЦП — цифровий підпис

ДПМ — довірений платформний модуль (англ. Trusted Platform Module, TMP)

ІСШДХ — інтегрована схема шифрування Діффі-Хеллмана (англ. Diffie-Hellman Integrated Encryption Scheme, DHIES)

ДБРЧП — доведення без розголошення Чаума-Педерсона (англ. Chaum-Pedersen Zero Knowledge Proof)

АЦПЕК — алгоритм ЦП на еліптичній кривій (англ. Elliptic Curve Digital Signature Algorithm, ECDSA)

ВСТУП

У цій роботі проводиться якісний і порівняльний аналіз методів забезпечення анонімності у найпоширеніших криптовалютах. Потім розписані всі етапи неінтерактивних стислих доведень без розголошення на прикладі протоколу GRO-16.

Будуть отримані результати у вигляді аналізу сценаріїв формування налаштування неінтерактивних стислих доведень без розголошення та ризиків, що з ними пов'язані. Також, отримані результати у вигляді розроблених атак, які можливі при таких сценаріях. Далі, запропоновані стратегії захисту від наведених атак при заданих сценаріях.

Актуальність дослідження визначається у бажанні, часто навіть необхідності, більшості людей та бізнесів зберігати анонімність своїх фінансів. Зокрема, виконанні нікому не підконтрольних та анонімних транзакцій в умовах повної недовіри.

Для цього є інструмент у вигляді неінтерактивних стислих доведеннях без розголошення, але це новий інструмент, який ще не добре вивчений. Не сформульовано умов, за яких ці доведення будуть стійкими. Немає аналізу умов, за яких вони будуть або не будуть стійкими. Не сформульовані рекомендації стосовно можливих вразливостей та підвищення стійкості, захисту від цих вразливостей.

Тому у розгляді цих питань і є актуальність даного дослідження.

Метою дослідження є аналіз вразливостей протоколу доведення без розголошення типу zk-SNARK GRO-16 та умов, за яких ці вразливості будуть критичними. Для досягнення мети необхідно розв'язати такі часткові завдання:

1) розглянути роботу протоколів zk-SNARK та всіх їх етапів на прикладі протоколу GRO-16;

2) запропонувати послідовність дій та алгоритми для формування елементів множини початкових налаштувань, для перевірки доведень для

протоколу GRO-16 та проаналізувати запропоновані алгоритми з точки зору стійкості протоколу GRO-16;

3) розробити атаки викрадання монет, які використовують знання зловмисником відомостей з початкових налаштувань, на протокол GRO-16;

4) проаналізувати загрози послаблення стійкості протоколу GRO-16 у сценарії розпаралелювання формування налаштування та запропонувати рекомендації щодо механізмів захисту від розроблених у цій роботі атак.

Об'єктом дослідження у даній роботі є процес захисту персональних даних при виконанні транзакцій у блокчейні.

Предметом дослідження є атаки на протоколи захисту персональних даних у блокчейні за умови повторного використання налаштування.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: абстрактна алгебра, теорія складності алгоритмів, алгебраїчна геометрія.

Наукова новизна отриманих результатів: вперше показано, які загрози несе повторне використання налаштування у протоколах неінтерактивних стислих доведень без розголошення та умови, які необхідні для цих атак; вперше запропоновані методи захисту від подібних атак; вперше поетапно розписано етап SETUP, на якому учасники блокчейну формують налаштування. Також, було вперше запропоновано використання довірених платформних модулів для посилення стійкості протоколів.

Практичне значення. Отримані результати є корисними як з точки зору зловмисника, так і з точки зору обґрунтування стійкості протоколів неінтерактивних стислих доведень без розголошення. Для зловмисника практична важливість у тому, що він може проаналізувати умови й одразу ж бачити, зможе він провести атаку чи ні. А з протилежної точки зору учасники можуть проаналізувати, чи є загрози в мережі та підвищити її стійкість, використовуючи запропоновані у роботі методи.

Апробація результатів та публікації. Частину результатів даної роботи було доповідано на таких конференціях: XVIII Науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики» (12-13 травня 2020 р., м. Київ).

1 ПРОБЛЕМИ ЗАБЕЗПЕЧЕННЯ АНОНІМНОСТІ У НАЙБІЛЬШ ПОШИРЕНИХ КРИПТОВАЛЮТАХ

У більшості звичайних людей та бізнесів існує бажання, часто навіть необхідність, зберігати анонімність своїх фінансів. Тому з появою у 2009 році першої криптовалюти Bitcoin (BTC) [1] у багатьох людей та організацій виникли сподівання, що тепер вони отримають способи виконання швидких, зручних, нікому непідконтрольних та анонімних транзакцій.

Дійсно, концепція BTC містила багато революційних для того часу ідей та технологічних рішень, найсуттєвішим з яких є спосіб досягнення децентралізації в умовах повної недовіри. Але з моменту запуску BTC та до цього часу алгоритм функціонування мережі практично не змінювався. Зокрема, не було зроблено нічого для створення та втілення додаткових механізмів анонімності. Проте на базі платформи BTC було створено ряд інших криптовалют (альткоїнів), які намагались тими чи іншими методами підвищити як анонімність користувачів, так і неможливість відстежувати обіг їх коштів.

Далі розглянуто основні способи забезпечення анонімності при користуванні різними криптовалютами та наведені їх переваги та недоліки.

Зазначимо, що анонімність отримувача коштів закінчується тоді, коли він виводить криптовалютні кошти на іменну банківську картку. Використання криптовалюти для оплати реальних товарів або послуг, як правило, вимагає її перетворення у валюту відповідної країни. Тому у цьому розділі в основному розглянуто способи збереження анонімності джерел, з яких криптовалюта надходить в електронний гаманець певної особи.

1.1 Способи збереження анонімності джерела надходження коштів

Якщо користувач сам створює свій гаманець, а не доручає його створення певній біржі [2], то створення гаманця не потребує ніякої особистої інформації про користувача. Більш того, він може створити довільну кількість гаманців так, що неможливо буде визначити, чи належать вони одній особі, чи різним. Сучасні програми-клієнти можуть автоматично створювати для користувача нові адреси гаманців для кожної транзакції, щоб його було складніше відстежити.

Але всі транзакції назавжди зберігаються у блокчейні, тобто інформація про пересилання коштів між гаманцями є загальнодоступною і її неможливо видалити. Тому якщо вдалось встановити зв'язок між якоюсь особою та певним гаманцем, то за інформацією, що зберігається у блокчейні, можна визначити всі пересування коштів у цей гаманець або з нього.

Крім того, якщо створювати гаманець на біржі, то досягнути анонімності ще важче. Деякі біржі вимагають певну особисту інформацію про клієнта, відповідно до законодавства тієї країни, до якої належить ця біржа. Сатоши Накамото, розробник Bitcoin, стверджував, що його криптовалюта є анонімною. Як стало зрозумілим пізніше, під цим визначенням він розумів можливість користувача не надавати ніяких особистих даних про себе для отримання цифрових грошей. Якби біткоїни добувались лише майнінгом, то криптовалюта дійсно була б анонімною. Але через певні правила, що встановлені на великих біржах, багато BTC-адрес можна пов'язати з реальними особистостями. До того ж, через властивість блокчейна зберігати всю інформацію щодо транзакцій, можна отримати фактичне підтвердження того, яку суму власник одного гаманця перевів власнику іншого. А також яка кількість грошей на вказаному рахунку.

Спеціалісти Bitfury заявили, що можуть розкрити особистості близько 16% всіх власників BTC-адрес [3]. А за кілька років до цього група розробників CryptoLux провела дослідження анонімності транзакцій в мережі BTC й довела, що можна успішно деанонімізувати до 60% адрес. У своїй роботі вони показали можливість прив'язки BTC-рахунку до IP-адрес користувачів, навіть якщо ті користуються мережею Tor або подібними VPN-клієнтами.

Це підштовхнуло багатьох розробників як до пошуку способів вдосконалення анонімності, так і до створення нових альткоїнів з потрібними властивостями, зокрема з підвищеною анонімністю.

1.1.1 Самостійний майнінг криптовалют

Найпростіший спосіб отримувати криптовалюти з анонімного джерела — це майнити їх самому. Таким чином у гаманці або з'являється нова, щойно створена монета, яка ще не була у жодному іншому гаманці, або монета переводиться з гаманця, що відповідає майнінговому пулу. Недоліки: накопичення монет у такий спосіб відбувається досить довго; вимагає постійної затрати ресурсів (електроенергії) та постійного підключення до Інтернету.

1.1.2 Використання міксерів транзакцій: Bitcoin-міксери та криптовалюта Dash

Загальний принцип робота міксера транзакцій полягає у наступному. На одному з етапів пересилання коштів виконується так звана колективна транзакція (або групова транзакція), в якій беруть участь відразу кілька платників. Монети, які вони пересилають, розбиваються на дрібні «шматочки», які перемішуються між собою, і відправляються отримувачам за певним алгоритмом. В результаті

отримувачі отримують правильну кількість монет, але ця сума буде складатись з різних шматочків, відправлених різними платниками. Механізм анонімізації повинен приховати однозначну відповідність між відправниками та їх монетами. Невизначеність полягає у тому, що отримані кошти могли з рівною імовірністю бути відправленими будь-яким з учасників колективної транзакції.

Саме за таким принципом влаштовані BTC-міксери [4] та міксери криптовалюти Dash (попередня назва – DarkCoin) [5]. Сучасні міксери можуть обслуговувати кілька видів криптовалют. Крім того, вони удосконалені додатковими функціями, такими як:

– перемішування не тільки адрес відправників, але й адрес отримувачів;

– затримка у часі доставлення монет (тобто переслана сума не тільки приходить частинками з інших адрес, але й ці частинки приходять у різний час);

– видалення інформації про транзакції через певний час після перемішування;

– інтеграція з анонімною мережею Tor;

– захист від «брудних» грошей.

Недоліком BTC-міксерів можна вважати те, що для міксування кошти користувача начебто передаються міксеру. Оскільки вихідні коди багатьох міксерів невідомі, то складно сказати, які там можуть бути приховані можливості (наприклад, міксер може запам'ятовувати «шлях» коштів під час перемішування, або зберігати якусь конфіденційну інформацію про користувача і потенційно її потім комусь надавати).

На відміну від BTC, криптовалюта Dash не потребує додаткової функції міксерів, оскільки ця функція перемішування в ній існує на рівні протоколу. Тому використання цих міксерів практично не збільшує час проведення транзакції, на відміну від використання BTC-міксерів.

Однією з переваг Dash у порівнянні з BTC-міксерами є те, що кошти під час перемішування залишаються власністю користувача. Наступною є

повністю відкритий код. Вихідний код як клієнтської частини (гаманців), так і мережевої інфраструктури (Мастернод) опубліковано для загального контролю на предмет відсутності прихованих функцій. Це ж саме стосується і механізмів анонімізації. Використовуючи вихідний код, користувачі можуть самостійно компілювати програму-гаманець, в якій будуть гарантовано відсутні незадокументовані можливості.

1.1.3 Використання анонімних платіжних систем

Альтернативами міксерів можуть бути спеціальні гаманці [6] з великим ступенем анонімності, наприклад, Electrum.

Також існують гаманці з вбудованою опцією перемішування монет. У 2014 році Коді Вілсон та Амір Тааки представили проєкт Dark Wallet —- додаток для браузера та клієнт для операційної системи Ubuntu. В основі Dark Wallet лежить метод CoinJoin, який теж є міксером транзакцій. Чим більше користувачів гаманця, тим більше можливостей для перемішування, а, отже, вищий ступінь анонімності.

Ще одним способом забезпечення анонімності транзакцій є використання спеціальних мереж. На сайті Darknetmarkets [7] можна знайти інструкцію, як зберегти анонімність при проведенні транзакцій у мережі Tor. Для цього користувачу потрібно зареєструвати кілька гаманців як в анонімній мережі, так і у відкритому сегменті інтернету, а також мати браузер Tor та сервіси-міксери, що підтримують Tor.

Дуже вдалим прикладом реалізації анонімної передачі криптовалют є платіжна система Z-Pay [8] з відкритим вихідним кодом. Основною особливістю цієї платіжної системи є емісія та передача знеособлених платіжних векселів (чеків) як оплату за товари та послуги. Чек в системі Z-Pay — це анонімне грошове зобов'язання, номіноване у криптовалюті або фіаті (реальній валюті). Номер чека є випадковою комбінацією з 35 цифр. При передачі такого номера чека як оплату, не розкриваються особисті дані, а сама оплата відбувається миттєво. При цьому отримувач

оплати може вивести кошти у будь-якій зручній для нього валюті. Таким чином, з використанням Z-Pay можна виконувати платежі, сплачувати будь-які послуги, обмінювати криптовалюту та фіатні кошти без верифікації.

До недоліків цієї платіжної системи можна віднести комісію при обміні та виводі коштів (1-3%). Також не зовсім зрозумілий принцип роботи системи, а сааме яка інформація про транзакцію зберігається та внаслідок чого забезпечується анонімність.

1.1.4 Купівля-продаж криптовалют через криптомати або біржі без верифікації

Найпростіший спосіб купівлі біткоїнів — через криптомат (спеціальний апарат для купівлі криптовалют, схожий на банкомат). Більшість криптоматів, крім BTC, підтримують також Ethereum, Bitcoin Cash, Litecoin, Dash, Dogecoin, а також анонімні ZCash й Monero, а деякі навіть більш як 40 різних криптовалют. Процедура купівлі дуже проста, але й комісія досить висока: 5-6% від суми операції з гривні та + 0.001 BTC комісія самої біткоїн-мережі.

Щоб скористуватись криптоматом, потрібно просканувати адресу свого гаманця та внести готівку, не менше за певну мінімальну суму. Обробка запиту становить не більше ніж 10 хвилин, після чого криптомонети з'являються у вашому гаманці.

Деякі «високорівневі» моделі криптоматів, наприклад, від компанії Genesis Coin, дозволяють не лише купувати криптовалюту, а й обмінювати її на фіатні гроші. На відміну від купівлі криптовалюти, її продаж через криптовалютний ATM займає більше часу, оскільки відбувається у два етапи. Користувачу потрібно виконати наступні дії.

1) Вказати суму криптовалюти, яку він планує продати.

2) Пройти процедуру верифікації (за потребою), на якій ми

зупинимось нижче.

3) Просканувати QR-код на екрані пристрою або на виданому чеку для переводу вказаної суми на гаманець оператора (час, відведений на цю процедуру, обмежений).

4) Зачекати підтвердження транзакції вузлами мережі криптовалют (як правило, достатньо двох підтверджень).

5) Просканувати QR-код на видачу готівки, після чого криптомат видасть фіатні гроші.

Розглянемо проблеми анонімності та можливої верифікації. По-перше, виходячи з опису протоколу роботи криптомату, він пов'язаний з деяким криптовалютним гаманцем (див. відео BATM Termanal Management у [9], «internal wallet» на 2 хв 19 сек). За адресою цього гаманця, взагалі кажучи, можливо прослідкувати цей криптомат як джерело поповнення коштів. І хоча особа того, хто вносив кошти, лишається невідомою, проте можна виявити його географічне положення.

Щодо анонімності отримувача коштів, то тут можливі різні варіанти. При обміні криптовалюти на фіатні гроші через криптомат верифікація може бути потрібною у таких випадках:

1) сума до обміну перевищує встановлений ліміт (проте в більшості таких випадків оператори просто не дозволяють обмінювати суми, які перевищують ліміт, шляхом встановлення обмежень);

2) вимоги обов'язкової верифікації, що встановлені в певних країнах, незалежно від суми операції.

Процедура верифікації може виконуватись різними способами, наприклад, шляхом фотографування з певним документом у руках, що підтверджує особу, або скануванням відбитків пальців, або шляхом зчитування даних платіжної карти, або через номер телефону.

Деякі біржі також допускають операції з криптовалютами без верифікації, або з обмеженою верифікацією, або з верифікацією за бажанням. Інколи навіть можна перевести без верифікації певну (як правило, обмежену) суму грошей у фіатну валюту. При цьому біржі, як

правило, намагаються не порушувати законодавство країни, в якій вони зареєстровані. Тому, в залежності від чинних законів, вони теж можуть обмежувати суму, яка не вимагає верифікації, або обмежити операції, наприклад, не надавати послуг з виведення фіатних грошей. Досить повний перелік бірж, які тою чи іншою мірою підтримують анонімність, та умов, на яких вони працюють, наведено у [10].

1.2 Принципи функціонування інших анонімних криптовалют, переваги та недоліки

Анонімні монети — це P2P платіжні системи з власною внутрішньою розрахунковою одиницею. Їх головна мета — забезпечити повну конфіденційність фінансових операцій за допомогою спеціальних технологій та криптографічних протоколів.

1.2.1 Безумовно анонімна криптовалюта Monero

Monero — це поки єдина монета, всі операції з якою є анонімними за замовчуванням. Для всіх інших криптовалют функцію анонімності потрібно додатково налаштовувати (як BTC-міксери) або підключати при проведенні транзакції (як у криптовалюті Dash).

З публічного блокчейну Monero можна дізнатись, чи існує та чи інша адреса рахунку, але неможливо перевірити баланс цього рахунку та отримати доступ до історії транзакцій, з ним пов'язаних. Анонімність Monero забезпечується наступними складними криптографічними засобами та механізмами:

– кільцевими конфіденційними транзакціями з використанням доведення діапазону та криптографічними підтвердженнями, які дозволяють приховувати величини коштів, що пересилаються у транзакції;

– неінтерактивними доведеннями без розголошень, які дозволяють перевіряти транзакцію без знання відправника та отримувача коштів, без знання кількості монет та без використання довіреної сторони;

– одноразовими адресами, та одноразовими відкритими ключами, які користувач повинен генерувати за допомогою криптографічних механізмів та які приховують шлях пересування коштів;

– кільцевими підписами, що дозволяють приховати, з якої саме адреси (з певної групи, що складається з 11 адрес) було відправлено кошти.

Ще однією безумовною перевагою є відкритий код цієї криптовалюти та високий ступінь довіри до її розробників у криптовалютному суспільстві.

До недоліків Monero можна віднести високі комісії за транзакції.

1.2.2 «Найматематичніша» криптовалюта ZCash

Криптовалюта ZCash була розроблена компанією Zerocoin Electric Coin Company та анонсована 20 січня 2016 року. Це криптовалюта з відкритим вихідним кодом, що безумовно є її перевагою. Всі транзакції ZCash публікуються у загальнодоступному ланцюжку блоків, але відправник, отримувач та сума транзакції залишаються приватними.

Криптовалюта ZCash для забезпечення анонімності транзакції вперше застосувала доведення без розголошення нового типу, яке називається zk-SNARK [11]. Абревіатура zk-SNARK розшифровується як «zero knowledge Succinct Non-interactive ARguments for Knowledge», тобто «неінтерактивні стиснені аргументи знання із нульовим розголошенням». Частина «неінтерактивні» означає, що стороні P, яка доводить володіння секретом, не потрібно напряму взаємодіяти з стороною V, що буде доведення перевіряти. А частина «із нульовим знанням» означає, що перевірка доведення повина переконувати V, що P дійсно володіє секретом, але не розкрити ніякої про цей секрет інформації. «Стиснені» означає, що доведення без розголошення можуть бути перевірені

протягом кількох мілісекунд, а довжина доведення не залежить від обсягу інформації, знання якої доводиться. Воно може становити всього, наприклад, 288 байтів для протоколу GRO-16 [14]. Зворотною стороною медалі таких стисненних доведень є початковий етам SETUP, який замінює довірену сторону. Під час цього етапу учасники створюють та використовують значення, які потім вони повинні тримати у секреті, щоб генерувати параметри, які будуть використовуватись для створення та перевірки доведень. Цей етап досить громіздкий: він вимагає достаньо великої кількості ресурсів та часу. Інколи він може важити понад ста гігабайт та генеруватися понад добу. Але створені у цьому етапі параметри можна використовувати, якщо хоча б один учасник є чесним. Таким чином, мається на увазі, що параметри не містять такої інформації, яка б дозволила підроблювати доведення та порушувати вагомість.

Загалом, усі zk-SNARK повинні мати три основні властивості, які також притаманні всім доведенням без розголошення.

1) *Повнота*: якщо P дійсно володіє секретом, то завжди існує спосіб це довести V.

2) *Вагомість*: якщо P не володіє секретом, але намагається довести V, що володіє, то імовірність його переконати або знехтовно мала, або її можна як завгодно близько наблизити до нуля.

3) *Нульове розголошення*: щоб V не робив у рамках протоколу взаємодії з P, він не дізнається ніякої інформації про його секрет.

Розглянемо два приклади доведень без розголошення.

Приклад 1.1. Класичний приклад «Печера нульового розголошення». Нехай Віктор — могутній воїн, а Пеггі — геніальна чародійка. Віктор хоче бути певен, що Пеггі знає закляття воскресіння, але Пеггі не може його розповісти, бо це заборонено Орденом Магів та Чародійок.

Посеред їх пригоди вони раптово натикаються на печеру, яка цілком раптово має вигляд, зображений на рисунку 1.1. І так раптово вийшло, що

ці двері можна відчинити тільки вимовленням цього закляття, після чого вони знов зачиняються.

До печери є звичайний вхід A та біля нього чарівні двері B, створені сліпим богом. Якщо Пеггі увійде в ці двері B, то вона випадково з'явиться в точці C або D рівноймовірно. Ці точки знаходяться занадто далеко, щоб Віктор зміг з точки A дійти до них, коли Пеггі буде відчиняти двері, та підслухати її.

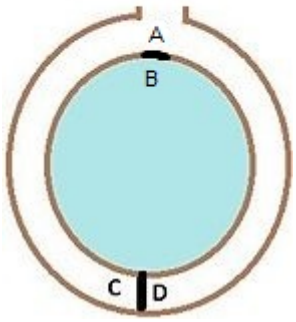

Рисунок 1.1 – Печера нульового розголошення

Пеггі запропонувала наступний протокол:

1) Віктор відправляється у точку A та стежить за виходом з печери;

2) Пеггі входить у двері B та випадково опиняється в точці C або D;

3) Віктор просить Пеггі прийти або с правої, або з лівої сторони.

Тобто, якщо Пеггі опинилась у точці D, а Віктор просить її прийти з правої сторони, то їй для цього необхідно відчинити двері, чого без знання закляття зробити неможливо.

Таким чином для Пеггі опинитись у потрібній точці складає $\frac{1}{2}$. Але цього може бути недостатньо, тому вони повторюють цей протокол стільки разів, поки Віктор не буде задоволений імовірністю везіння (тобто імовірністю обману), яка майже дорівнює нулю вже при повторенні в 20 разів: $\frac{1}{2^{20}} = 0.000000953$.

Приклад 1.2. Нехай, під час мандрування, з Пеггі стався

нещасний випадок. Віктор просить сліпого бога воскресити її, але сліпий бог відповідає, що зробить це, якщо Віктор зможе йому довести існування інших кольорів, окрім кольору темряви. У Віктора було з собою червоне та зелене яблуко. Він запропонував сліпому богу такий протокол:

1) сліпий бог повинен сховати два яблука за спину, перекласти з одної руки в іншу, або залишити як є;

2) він показує два яблука Віктору;

3) Віктор повинен відповісти, переклав сліпий бог яблука, чи ні.

Оскільки сліпий бог нікуди не поспішає, вони повторюють протокол 100 разів, зменшуючи імовірність обману до $\frac{1}{2^{100}} \approx 10^{-33}$.

Ці приклад протоколу є інтерактивним, тому що Пеггі (Віктору) потрібно напряму взаємодіяти з Віктором (сліпим богом), що довести свій секрет. Але вони демонструють, як працюють основні три властивості доведень без розголошення. Легко перевірити, що вони дійсно виконуються.

Протоколи zk-SNARK дозволяє перевіряти коректність і легітимність транзакції не розкриваючи ніякої інформації про:

1) відправника;

2) суму переводу;

3) отримувача.

Публічними є лише часові мітки.

Зауважимо, що Monero також використовує доведення без розголошення, але зовсім іншого типу, не такі універсальні та не такі математично складні. Перевага zk-SNARK полягає не тільки в універсальності, але й у меншому об'ємі доведенні.

Таким чином, zk-SNARK є важливою частиною забезпечення анонімності у блокчейні.

Окрім ZCash протоколи zk-SNARK знайшли своє місце в інших галузях:

– доведення тверджень на основі приватних даних:

– сутність A має більше ніж X на її банківському акаунт;

– за останній рік банк не проводив транзакцій із сутністю Y;

– порівняння ДНК без розголошення повного ДНК;

– сутність має кредит більше, ніж Z;

– анонімна авторизація:

– доведення, що запрошувач R має права для доступу закритої зони вебсайт без розголошення його особистості (наприклад, логін, пароль);

– доведення, що персона знаходиться у списку дозволенних країн/штату без розголошення з якої само;

– доведення, що персона володіє проїзний на метро без розголошення id;

– анонімні платежі:

– платежі із повним відстороненням від будь-якого типу особистих даних;

– платежі податків без розголошення доходів;

– Надійні обчислення:

– виконати доволі дороге обчислення та підтвердити, що результат коректний без повторення обчислення;

1.3 Початкові налаштування

У цьому підрозділі буде розглянуто один із типів відображень, який часто використовується у zk-SNARK. Також, розглянуто основні «важкі задачі», на які часто спираються при доведенні стійкості протоколів zk-SNARK.

1.3.1 Білінійні відображення

Нехай $\mathbf{G_1}$, $\mathbf{G_2}$, $\mathbf{G_3}$ — циклічні мультиплікативні групи простого порядку p (не обов'язково однакові). Нехай G_i — генератор групи $\mathbf{G_i}$, $i = \overline{1,\ 3}$.

Нехай відомо $G_i^a,\ G_i^b$ і невідомо $a,\ b \in \mathbb{Z}_p^*$, але потрібно обчислити $G_i^{ab},\ i = \overline{1,2}$. Оскільки ця задача є складною по припущенню Діффі-Хеллмана, невідомо, як це зробити без знання a або b, але значення G_i^{ab} може бути потрібно. Білінійні відображення використовуються, зокрема, для розв'язання цієї проблеми у деякому сенсі. Це такі доволі специфічні важкооборотні функції, які доволі часто використовуються у доведеннях без розголошення.

Означення 1.1. *Білінійним відображенням* $e\colon \mathbf{G_1} \times \mathbf{G_2} \to \mathbf{G_3}$ називається таке відображення, що:

1) $\forall u \in \mathbf{G_1},\ v \in \mathbf{G_2},\ a, b \in \mathbb{Z}_p^* : e(u^a, v^b) = e(u, v)^{ab}$.

2) Існує алгоритм, що обчислює функцію e за поліноміальний час від довжини вхідних даних.

3) Відображення не повинно бути виродженим, тобто $e(G_1, G_2) \not\equiv 1_{\mathbf{G_3}}$, де $1_{\mathbf{G_3}}$ — нейтральний елемент групи $\mathbf{G_3}$.

1.3.2 Важкі задачі

Доволі часто стійкість доведень без розголошення обумовлюється «важкістю» деяких задач, тобто таких, для яких невідомо алгоритм, що вирішують їх за поліноміальний час від довжини вхідних даних. Нижче наведено важкі задачі, які зазвичай використовуються. Зокрема, у цій роботі.

Об'єкти $\mathbf{G_1},\ \mathbf{G_2},\ \mathbf{G_3},\ G_1,\ G_2,\ G_3,\ p,\ e(\cdot,\cdot)$ ті ж самі, що й у попередньому підрозділі. Вони вважаються відомими.

Припущення про складність обчислювальної задачі Діффі-Хеллмана.

Якщо відомо $G_i^a,\ G_i^b$ і невідомо $a,\ b \in \mathbb{Z}_p^*$, то важко обчислити G_i^{ab}, $i = \overline{1,2}$.

Припущення про складність розрізнювальної задачі Діффі-Хеллмана.

Якщо відомо $G_i^a,\ G_i^b,\ G_i^c$ і невідомо $a,\ b,\ c \in \mathbb{Z}_p^*$, то важко обчислити значення $e(G_1, G_2)^{abc}$ та відрізнити його від випадкового $Z \in \mathbf{G_3}$, $i = \overline{1,2}$.

Білінійне допущення створення пар.

Якщо відомо $e(X, G_2)$ або $e(G_1, X)$, то важко обчислити X.

Білінійне допущення Діффі-Хеллмана.

Якщо відомо $G_i^a,\ G_i^b$ і невідомо $a,\ b \in \mathbb{Z}_p^*$, то важко обчислити $e(H,\ G_2)^{ab}$ для випадкового $G_1 \neq H \in \mathbf{G_1}$.

Висновки до розділу 1

У цьому розділі було розглянуто різні способи забезпечення анонімності в криптовалютах. Було проведено порівняльний аналіз цих методів. Знайдено переваги та недоліки кожного методи. В результаті аналізу показано, що анонімність може бути забезпечена як організаційними заходами, типу біткоїн міксерів або Dash міксерів, так і математичними способами, як, наприклад, у криптовалюті ZCash. Наприклад, міксери валют простіші для розуміння та реалізації, але, якщо немає відкритого коду, то користувачі не можуть бути впевнені, що міксери працюють чесно та не збирають персональних даних. zk-SNARK гарантують забезпечення анонімності на рівні протоколу, але вони потребують складного математичного апарату та складної практичної реалізації.

2 ПРАКТИЧНА РЕАЛІЗАЦІЯ ПРОТОКОЛУ GRO-16

Результати, наведені у даному розділі, були доповідані автором у [13].

У даному розділі розглянуті практичні аспекти реалізації zk-SNARK протоколу GRO-16, які не були наведені в оригінальній статті [14]. Більш детально розглянуто питання генерації налаштування на етапі SETUP, послідовність дій у генерації, алгоритми, тощо. Деякі моменти можуть повторювати оригінальну статтю, але це необхідно для створення цілісної картини реалізації протоколу та атак на нього.

2.1 Постановка задачі

Розглядається схема блокчейна, у якій кожний учасник володіє деякою кількістю одиниць криптовалюти — *монетами*, кожна з яких має власний унікальний набір значень $\{a_i\}_{i=0}^{m}$, який розбивається на дві підмножини: підмножина-ідентифікатор (далі, ідентифікатор) монети $\{a_i\}_{i=0}^{l}$, яка відповідає можливості однозначно ідентифікувати монету та відома усім учасникам; підмножина-таємниця (далі, таємниця) монети $\{a_i\}_{i=l+1}^{m}$, знання якої означає володіння цією монетою, тобто повинна бути відома тільки її власнику. Зрозуміло, що для виконання транзакцій з монетами необхідно бути власником монети, тобто, знати її секрет.

Нехай у нульовому блоці блокчейна опубліковані елементи $u_{i,q},\ v_{i,q},\ w_{i,q} \in \mathbb{F}_p,\ i = \overline{0,\ m},\ q = \overline{0,\ n}$, де $\mathbb{F}_p$ — поле великої простої характеристики p. Ці значення будуть однакові в усій цій роботі. У наступних блоках доведення володіння монетою зводиться до задачі доведення без розголошення знання таких $\{a_i\}_{i=0}^{m}$, $a_0 = 1$, що:

$$\sum_{i=0}^{m} a_i u_{i,q} \cdot \sum_{i=0}^{m} a_i v_{i,q} = \sum_{i=0}^{m} a_i w_{i,q},\ q = \overline{0,n}. \tag{2.1}$$

Переформулюємо задачу (2.1) у термінах поліномів.

Побудуємо $3(m+1)$ інтерполяційних поліномів Лагранжа,

$$u_i(x),\ v_i(x),\ w_i(x) \in \mathbb{F}_p[X],\ i = \overline{0,\ m} \tag{2.2}$$

з обмеженнями

$$u_i(q) = u_{i,q},\ v_i(q) = v_{i,q},\ w_i(q) = w_{i,q},\ i = \overline{0,\ m},\ q = \overline{1,\ n}. \tag{2.3}$$

Зауважимо, що ці поліноми мають степінь не більше, ніж $n-1$. Означимо поліном

$$g(x) = \sum_{i=0}^{m} a_i u_i(x) \cdot \sum_{i=0}^{m} a_i v_i(x) - \sum_{i=0}^{m} a_i w_i(x),$$

де $\{1, 2, ..., n\}$ — корені полінома $g(x)$, згідно з (2.1), (2.2), (2.3). За теоремою Безу

$$g(x) \vdots (x-q),\ q = \overline{1,\ n}.$$

Оскільки усі поліноми $(x-q),\ q = \overline{1,\ n}$, незвідні та попарно взаємно прості, то відповідно до наслідку алгоритму Евкліда:

$$g(x) \vdots \prod_{q=1}^{n} (x-q).$$

Нехай $t(x) = \prod_{q=1}^{n} (x-q)$. Тоді задачу (2.1) можемо переформулювати так: довести без розголошення знання таких $\{a_i\}_{i=0}^{m}$, що

$$\sum_{i=0}^{m} a_i u_i(x) \cdot \sum_{i=0}^{m} a_i v_i(x) \equiv \sum_{i=0}^{m} a_i w_i(x) \pmod{t(x)},$$

або

$$\sum_{i=0}^{m} a_i u_i(x) \cdot \sum_{i=0}^{m} a_i v_i(x) = \sum_{i=0}^{m} a_i w_i(x) + h(x)t(x),$$

$$x \in \{1, 2, ..., n\}. \tag{2.4}$$

2.2 Формування налаштування

Як зазначалось у попередньому розділі, у протоколах zk-SNARK на етапі SETUP формується множина початкових значень — *налаштування*, яка замінює довірену сторону та використовується для формування доведень та їх перевірки будь-якими учасниками протоколу. Зазвичай, налаштування складається із таких значень, у формуванні яких беруть участь усі учасники блокчейну, яле жоден з них не знає про ці значення, оскільки вони відповідають за вагомість zk-SNARK, що буде показано у наступному розділі. Проте, ця умова та вагомість zk-SNARK зберігається, якщо є хоча б один чесний учасник.

Розглянемо основні об'єкти, які фігурують в цій роботі. Отже, нехай:

– $\mathbf{G_1}$ — циклічна підгрупа великого простого порядку p деякої еліптичної кривої $E(\mathbb{F}_r)$ над простим полем $\mathbb{F}_r$;

– $\mathbf{G_2}$ — циклічна підгрупа великого простого порядку p тієї ж еліптичної кривої $E(\mathbb{F}_{r^k})$ над деяким розширенням $\mathbb{F}_{r^k}$;

– $\mathbf{G_3}$ — циклічна підгрупа великого простого порядку p групи $\mathbb{F}_{r^k}^*$;

– $e\colon \mathbf{G_1} \times \mathbf{G_2} \to \mathbf{G_3}$ — білінійне відображення;

– $G_1,\ G_2$ — генератори груп $\mathbf{G_1},\ \mathbf{G_2}$ відповідно.

Тут і надалі «+» позначає операції у $\mathbf{G_1}$ та $\mathbf{G_2}$. Відповідно «$*$» позначає операцію в $\mathbf{G_3}$.

Нехай $a,\ b \in \mathbb{F}_p,\ G \in \mathbf{G_1} \cup \mathbf{G_2}$, тоді цілком природно прийняти

$$aG = \underbrace{G + G + \cdots + G}_{a \text{ разів}},$$

$$(a+b)G = aG + bG,$$

$$0 \cdot G = \mathbf{O}.$$

Задача формування налаштування на етапі SETUP у протоколі GRO-

16 полягає в обчисленні елементів множини

$$\{G_\alpha^{(1)}, G_\beta^{(1)}, G_\delta^{(1)}, \{X_i^{(1)} = x^i G_1\}_{i=0}^{n-1}, \{G_{\alpha\beta\gamma,i}^{(1)}\}_{i=0}^{l}, \{G_{\alpha\beta\delta,i}^{(1)}\}_{i=l+1}^{m}, \{T_{\delta,i}^{(1)}\}_{i=0}^{n-2}\}, \tag{2.5}$$

та елементів множини

$$\{G_\beta^{(2)}, G_\gamma^{(2)}, G_\delta^{(2)}\{X_i^{(2)} = x^i G_2\}_{i=0}^{n-1}\}, \tag{2.6}$$

де

$$\begin{aligned} G_\rho^{(j)} &= \rho G_j,\ \rho \in \{\alpha, \beta, \delta, \gamma\},\ j = \overline{1,\ 2}, \\ G_{\alpha\beta\gamma,i}^{(1)} &= \frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\gamma} G_1, \\ G_{\alpha\beta\delta,i}^{(1)} &= \frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\delta} G_1, \\ T_{\delta,i}^{(1)} &= \frac{x^i t(x)}{\delta} G_1. \end{aligned}$$

Означення 2.1. Елементи $\alpha,\ \beta,\ \gamma,\ \delta,\ x^i \in \mathbb{F}_p,\ i = \overline{1, n}$ називаються *критичними елементами налаштування.*

Побудувати елементи множин (2.5) та (2.6) потрібно так, щоб виконувались такі вимоги:

– усі знали кожне значення елементів цих множин;

– кожний учасник бере участь в обчисленні елементів цих множин;

– якщо якийсь учасник діє не по протоколу, усі інші учасники це бачать;

– жоден учасник не знає про критичні елементи налаштування.

Нехай запропоновано наступний алгоритм формування перших трьох елементів множин (2.5) та (2.6) на прикладі обчислення $G_\alpha^{(1)} = \alpha G_1$. Після його опису розглянемо, чи відповідає він вищезазначеним умовам.

Алгоритм 2.1. Обчислення $G_\alpha^{(1)} = \alpha G_1$ k учасниками протоколу.

1) Учасник P_1 обраховує значення $U_1 = \alpha_1 G_1$ та публікує U_i у блокчейн, де $\alpha_1 \in \mathbb{F}_p$ обрано випадково.

2) Послідовно кожний учасник P_i, починаючи з $i \geq 2$, обчислює значення $U_{1,2,\ldots,i} = \alpha_i U_{1,2,\ldots,i-1}$ та публікує $U_{1,2,\ldots,i}$ у блокчейн.

3) Встановлюється $G_\alpha^{(1)} = U_{12\ldots k}$.

4) $G_\alpha^{(1)}$ публікується у блокчейн.

Означення 2.2. Елементи $\alpha_i,\ \beta_i,\ \gamma_i,\ \delta_i,\ x_i$, які випадково обирає учасник $P_i,\ i = \overline{1,n}$ називаються *елементами розкладу учасника* P_i.

Цей алгоритм може бути зручним через простоту своєї реалізації, але він не відповідає необхідним умовам. У ньому відсутня перевірка того, що учасник дійсно множить на свій елемент розкладу, а не просто публікує якесь випадкове значення у блокчейн. Така публікація може, по-перше, анулювати участь попередніх учасників. По-друге, останній учасник P_k може просто помножити G_1 на свій елемент розкладу та опублікувати у блокчейн. І вийде, що $\alpha G_1 = \alpha_k G_1$. А це одразу ж (буде показано у наступному розділі) порушить вагомість протоколу GRO-16.

Тому при виборі алгоритму потрібно враховувати ситуацію повної взаємної недовіри. Розглянемо модифікацію цього алгоритму із використання білінійних відображень.

Алгоритм 2.2. Обчислення $G_\alpha^{(1)} = \alpha G_1$ k учасниками протоколу.

1) Учасник P_i, обирає свій елемент розкладу $\alpha_i \in \mathbb{F}_p$, $i = \overline{1,k}$, який тримає у секреті.

2) Учасник P_i обчислює $U_i = \alpha_i G_1,\ T_i = \alpha_i G_2$ та публікує $\{U_i,\ T_i\}$ у блокчейн, $i = \overline{1,k}$.

3) Учасник P_j, перевіряє рівність

$$e(U_i, G_2) = e(G_1, T_i),\ i = \overline{1,k},\ \ j = \overline{1,k},\ i \neq j.$$

Видно, що якщо учасник P_i чесний, то, відповідно до властивостей білінійного відображення у 1.3.2,

$$e(U_i,\ G_2) = e(\alpha_i G_1,\ G_2) = e(G_1,\ \alpha_i G_2) = e(G_1, T_i).$$

Враховуючи білінійне допущення створення пар, важко, наприклад, маючи деяке випадкове значення U_i та $e(U_i, G_2)$ обчислити значення T_i таке, що $e(U_i,\ G_2) = e(G_1, T_i)$. Можна вважати, що виконання цієї рівності свідчить про те, що P_i дійсно обчислив значення з множини $\{U_i,\ T_i\}$ з тим самим a_i. Якщо i-та рівність не виконується, учасник P_i є нечесним та має понести покарання.

4) Коли всі перевірки у попередньому кроці виконані, учасник P_i обчислює значення $U_{1,2,\ldots,i} = \alpha_i U_{1,2,\ldots,i-1}$ та публікує $U_{1,2,\ldots,i}$ у блокчейн, $i = \overline{2,k}$.

5) У той самий час, з кожною публікацією учасником P_i значення $U_{1,2,\ldots,i}$, учасник P_j перевіряє рівність

$$e(U_{1,2,\ldots,i}, G_2) = e(U_{1,2,\ldots,i-1}, T_i),\ i = \overline{1,k},\ \ j = \overline{1,k},\ i \neq j.$$

Аналогічно, якщо i-та рівність не виконується, учасник P_i є нечесним та має понести покарання.

6) Якщо всі учасники дотримувались протоколу, встановлюється $G_\alpha^{(1)} = U_{1,2,\ldots,k}$.

Отже, легко перевірити, що цей алгоритм вже відповідає необхідним умовам. Доцільно вважати, що жоден з учасників не вміє розв'язувати задачу дискретного логарифмування на еліптичних кривих.

Зауваження. Обчислення множин $\left\{X_i^{(1)}\right\}$ та $\left\{X_i^{(2)}\right\}$, $i = \overline{1,\ n-2}$ трохи відрізняється від алгоритму 2.2, але ідея така ж сама: обчислення та перевірка за допомогою білінійних відображень. Кожна генерація $x^i G_j$ відбувається як $x \cdot x^{i-1} G_j = x \cdot X_{i-1}^{(j)}$, $j = \overline{1,\ 2}$.

Розглянемо, як учасники можуть обчислити, наприклад, $u_i(x)G_1$. Нехай

$$u_i(x) = A_{n-1}^{(i)} x^{n-1} + A_{n-2}^{(i)} x^{n-2} + \cdots + A_1^{(i)} x + A_0^{(i)}.$$

Тоді

$$\sum_{j=0}^{m-1} A_j^{(i)} x^j G_1 = \sum_{j=0}^{m-1} A_j^{(i)} X_j^{(1)} = u_i(x) G_1.$$

Коли настає час обчислювати $\left\{G^{(1)}_{\alpha\beta\gamma,i}\right\}_{i=0}^{l}$ та $\left\{G^{(1)}_{\alpha\beta\delta,i}\right\}_{i=l+1}^{m}$, то потрібно дотримуватись наступних міркувань. Значення $\beta u_i(x)G_1$ та $\alpha v_i(x)G_1$ потрібно зробити невідомими для кожного учасника протоколу, але зробити відомими значення $\frac{\beta u_i(x)}{\gamma}G_1$, $\frac{\alpha v_i(x)}{\gamma}G_1$, $\frac{\beta u_i(x)}{\delta}G_1$ та $\frac{\alpha v_i(x)}{\delta}G_1$. Це не вплине на формування налаштування, тому що

$$\frac{\beta u_i(x)}{\gamma}G_j + \frac{\alpha v_i(x)}{\gamma}G_j + \frac{w_i(x)}{\gamma}G_j = \frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\gamma}G_j,$$

де $i = \overline{0,\ m},\ j = \overline{1,\ 2}$.

Аналогічно

$$\frac{\beta u_i(x)}{\delta}G_j + \frac{\alpha v_i(x)}{\delta}G_j + \frac{w_i(x)}{\delta}G_j = \frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\delta}G_j,$$

де $i = \overline{0,\ m},\ j = \overline{1,\ 2}$.

І потрібно дотримуватись саме цієї послідовності тому, що витік інформації про значення $\beta u_i(x)G_1$ та $\alpha v_i(x)G_1$ можна використати для побудови атак на GRO-16 таким чином, щоб порушити його вагомість. Як це можна зробити, показано у наступному розділі. Наступний алгоритм демонструє ідею, як схожі значення обчислюються.

Нехай k учасників протоколу використали алгоритм 2.2 для побудови $G^{(2)}_{\gamma} = \gamma G_2$. Таким чином, кожному учаснику P_i вже відповідає значення $\gamma_i \in \mathbb{F}_p$ і множина $\{D_i,\ T_i\}$, де

$$D_i = \gamma_i G_1,$$
$$T_i = \gamma_i G_2,$$
$$i = \overline{1, k}.$$

Нехай k учасників протоколу використовують алгоритм 2.2 для побудови $G^{(1)}_{\beta} = \beta G_1$. Таким чином, кожному учаснику P_i вже відповідає

значення $\beta_i \in \mathbb{F}_p$ і множина $\{H_i,\ L_i\}$, де

$$H_i = \beta_i G_1,$$
$$L_i = \beta_i G_2,$$
$$i = \overline{1,k}.$$

Нехай k учасників протоколу використали алгоритм 2.2 для побудови $\{X_i^{(1)}\}_{i=0}^{n-1}$.

Алгоритм 2.3. Обчислення $\left\{\frac{\beta u_i(x)}{\gamma} G_1\right\}_{i=0}^{l}$ k учасниками протоколу.

1) k учасників протоколу обчислюють $U_i = u_i(x)G_1$, $i = \overline{0,l}$.

2) Учасник P_i обчислює значення $\gamma_i^{-1} \bmod p$, $i = \overline{1,k}$.

3) Учасник P_i обчислює значення $Z_i = \gamma_i^{-1} G_2$ і публікує Z_i у блокчейн, $i = \overline{1,k}$.

4) Учасник P_j перевіряє рівність

$$e(D_i, Z_i) = e(G_1, G_2),\ i = \overline{1,k},\ \ j = \overline{1,k},\ i \neq j.$$

5) Учасник P_i обчислює значення $J_{i,j} = \gamma_i^{-1} J_{i-1,j}$ і публікує $\{J_{i,j}\}_{j=0}^{l}$, де $J_{0,j} = U_j$, $i = \overline{1,k}$.

6) У той самий час, з кожною публікацією учасником P_i значень $\{J_{i,j}\}_{j=0}^{l}$, учасник P_h перевіряє рівність

$$e(J_{i,j}, T_i) = e(J_{i-1,j}, G_2),\ i = \overline{1,k},\ \ h = \overline{1,k},\ i \neq h.$$

7) Якщо жоден учасник не порушував протокол, то встановлюється і публікується $\{Q_i = J_{k,i}\}_{i=0}^{l}$.

8) Послідовно, кожний учасник P_i обчислює значення $R_{i,j} = \beta_i R_{i-1,j}$ і публікує $\{R_{i,j}\}_{j=0}^{l}$, де $R_{0,j} = Q_j$, $i = \overline{1,k}$.

9) У той самий час, з кожною публікацією учасником P_i значень

$\{R_{i,j}\}_{j=0}^{l}$, учасник P_h перевіряє рівність

$$e(R_{i,j}, G_2) = e(R_{i-1,j}, H_i),\ i = \overline{1,k},\ \ h = \overline{1,k},\ i \neq h.$$

10) Якщо жоден учасник не порушував протокол, то встановлюється $\frac{\beta u_i(x)}{\gamma} G_1 = R_{k,i}$ і публікується $\{\frac{\beta u_i(x)}{\gamma} G_1\}_{i=0}^{l}$.

Множина значень $\left\{G^{(1)}_{\alpha\beta\delta,i}\right\}_{i=l+1}^{m}$ обчислюється аналогічно. Також аналогічним чином будується $\left\{T^{(1)}_{\delta,i}\right\}_{i=0}^{n-2}$.

Отже, після довгих обчислень нарешті маємо дві множини:

$$\sigma_1 G_1 = \{G^{(1)}_{\alpha}, G^{(1)}_{\beta}, G^{(1)}_{\delta}, \{X^{(1)}_i\}_{i=0}^{n-1}, \{G^{(1)}_{\alpha\beta\gamma,i}\}_{i=0}^{l},$$
$$\{G^{(1)}_{\alpha\beta\delta,i}\}_{i=l+1}^{m}, \{T^{(1)}_{\delta,i}\}_{i=0}^{n-2}\},$$
$$\sigma_2 G_2 = \{G^{(2)}_{\beta}, G^{(2)}_{\gamma}, G^{(2)}_{\delta}\{X^{(2)}_i\}_{i=0}^{n-1}\}.$$

Означення 2.3. *Налаштуванням* називається множина

$$Setup = \{\sigma_1 G_1, \sigma_2 G_2\}. \tag{2.7}$$

2.3 Формування доведення

Розглянемо, як будуються доведення у протоколі GRO-16.

Нехай учасник P володіє монетою зі значеннями $\{a_i\}_{i=0}^{m}$. Він хоче це довести всім іншим учасникам і провести з нею деяку транзакцію. Для цієї задачі він використовує алгоритм формування доведення того, що він володіє монетою з ідентифікатором $\{a_i\}_{i=0}^{l}$. Як було зазначено раніше, якщо він дійсно володіє цією монетою, то йому відома її таємниця $\{a_i\}_{i=l+1}^{m}$.

Означення 2.4. *Доведенням володіння монетою з ідентифікатором $\{a_i\}_{i=0}^{l}$ у zk-SNARK протоколі GRO-16 із дійсним на даний час Setup 2.7* (далі, *доведення*) називається четвірка

$\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$, де

$$A^{(1)} = AG_1,\ B^{(2)} = BG_2,\ C^{(1)} = CG_1,$$

$$A = \alpha + \sum_{i=0}^{m} a_i u_i(x) + r\delta,$$

$$B = \beta + \sum_{i=0}^{m} a_i v_i(x) + s\delta,$$

$$C = \frac{\sum_{i=l+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} + \frac{h(x)t(x)}{\delta} + sA + rB + rs\delta,$$

де $r,\ s \in \mathbb{F}_p$ обрані випадково учасником P і відомі тільки йому.

Розглянемо більш детально, як P обчислює трійку $\langle A^{(1)}, B^{(2)}, C^{(1)} \rangle$.

$$A^{(1)} = AG_1 = \alpha G_1 + \sum_{i=0}^{m} a_i u_i(x) G_1 + r\delta G_1 =$$

$$= G_\alpha^{(1)} + \sum_{i=0}^{m} a_i \sum_{j=0}^{n-1} A_{i,j} X_j^{(1)} + r G_\delta^{(1)},$$

де $A_{i,j}$ — коефіцієнти полінома $u_i(x)$.

$$B^{(2)} = BG_2 = \beta G_2 + \sum_{i=0}^{m} a_i v_i(x) G_2 + s\delta G_2 =$$

$$= G_\beta^{(2)} + \sum_{i=0}^{m} a_i \sum_{j=0}^{n-1} B_j X_j^{(2)} + s G_\delta^{(2)},$$

де $B_{i,j}$ — коефіцієнти полінома $v_i(x)$.

$$C^{(1)} = CG_1 = \frac{\sum_{i=l+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} G_1 +$$

$$+ \frac{h(x)t(x)}{\delta} G_1 + sAG_1 + rBG_1 - rs\delta G_1 =$$

$$= \sum_{i=l+1}^{m} a_i G_{\alpha\beta\delta,i}^{(1)} + \sum_{i=0}^{n-2} h_i T_{\delta,i}^{(1)} + sA^{(1)} + rB^{(1)} - rsG_\delta^{(1)},$$

де h_i — коефіцієнти полінома $h(x)$ із (2.4), який, відповідно, відомий тільки

власнику монети.

Зауваження. Навіть учасник P, який генерує доведення, не знає значення $A,\ B,\ C$.

Таким чином, введемо алгоритм формування доведення.

Алгоритм 2.4. Формування четвірки $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$ учасником P.

1) P випадково обирає два значення $r, s \in \mathbb{F}_p^*$ та тримає їх у секреті.

2) P обчислює

$$A^{(1)} = G_{\alpha}^{(1)} + \sum_{i=0}^{m} a_i \sum_{j=0}^{n-1} A_{i,j} X_j^{(1)} + r G_{\delta}^{(1)},$$

де $A_{i,j}$ — коефіцієнти полінома $u_i(x)$.

3) P обчислює

$$B^{(2)} = G_{\beta}^{(2)} + \sum_{i=0}^{m} a_i \sum_{j=0}^{n-1} B_j X_j^{(2)} + s G_{\delta}^{(2)},$$

де $B_{i,j}$ — коефіцієнти полінома $v_i(x)$.

4) P обчислює

$$C^{(1)} = \sum_{i=l+1}^{m} a_i G_{\alpha\beta\delta,i}^{(1)} + \sum_{i=0}^{n-2} h_i T_{\delta,i}^{(1)} + s A^{(1)} + r B^{(1)} + rs G_{\delta}^{(1)},$$

де h_i — коефіцієнти полінома $h(x)$ із (2.4).

5) P публікує доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)}, \{a_i\}_{i=0}^{l}\ \rangle$.

Зауважимо, що учаснику P для формування доведення не потрібно знати усі значення із *Setup* (2.7).

Означення 2.5. Множина елементів із *Setup*, яка використовується для формування доведення, називається *ключем доведення*:

$$PK = \{G_{\alpha}^{(1)},\ G_{\delta}^{(1)},\ \{X_i^{(1)}\}_{i=0}^{n-1},\ \{G_{\alpha\beta\gamma,i}^{(1)}\}_{i=0}^{l},\ \{T_{\delta,i}^{(1)}\}_{i=0}^{n-2},\ G_{\beta}^{(2)},\ G_{\delta}^{(2)},\ \{X_i^{(2)}\}_{i=0}^{n-1}\}.$$

Усім властивостям, які притаманні всім доведенням без розголошення, тобто, повнота, вагомість та нульове розголошення, протокол GRO-16 відповідає. Відповідні доведення щодо дотримання протоколом цих властивостей можна знайти в оригінальній статті [14].

2.4 Перевірка доведення

Тепер покажемо, як будь-який учасник V може перевірити опубліковану четвірку у якості доведення. Нагадаємо, що всім учасникам відомо $p,\ \mathbf{G}_1,\ G_1,\ \mathbf{G}_2,\ G_2$ та поліноми $u_i(\cdot),\ v_i(\cdot),\ w_i(\cdot),\ i = \overline{0,\ m}$ і $Setup$ (2.7).

Загалом, для перевірки кожного доведення, для формування якого використовувався один і той же $Setup$, можна перед обчислити наступні значення:

$$V_{\alpha\beta} = e(G_{\alpha}^{(1)}, G_{\beta}^{(2)}),$$
$$V_{\alpha\beta\gamma,i} = e(G_{\alpha\beta\gamma,i}^{(1)}, G_{\gamma}^{(2)}),\ i = \overline{0,l}.$$

Отже, коли якийсь учасник P опублікував четвірку $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$, учасник V обчислює такі значення:

$$V_{AB} = e(A^{(1)}, B^{(2)}),$$
$$V_{C\delta} = e(C^{(1)}, G_{\delta}^{(2)}).$$

І перевіряє рівність

$$V_{AB} = V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i V_{\alpha\beta\gamma,i}\right) V_{C\delta}. \tag{2.8}$$

Якщо рівність виконується, то учасник P дійсно володіє монетою з

ідентифікатором $\{a_i\}_{i=0}^{l}$, тобто знає її таємницю. Чому це так — буде показано в наступному підрозділі. Наразі введемо алгоритм перевірки доведення.

Алгоритм 2.5. Перевірка учасником V доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$ учасника P.

1) V обчислює

$$V_{AB} = e(A^{(1)}, B^{(2)}).$$

2) V обчислює

$$V_{C\delta} = e(C^{(1)}, G_{\delta}^{(2)}).$$

3) V перевіряє рівність

$$V_{AB} = V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i V_{\alpha\beta\gamma,i}\right) V_{C\delta}.$$

Якщо рівність виконується, то учасник P дійсно володіє монетою з ідентифікатором $\{a_i\}_{i=0}^{l}$. В іншому випадку, доведення не є дійсним.

Означення 2.6. Множина значень із *Setup*, які використовує V для перевірки доведення, називається *ключем перевірки*:

$$VK = \{G_{\alpha}^{(1)},\ G_{\beta}^{(2)},\ \{G_{\alpha\beta\gamma,i}^{(1)}\}_{i=0}^{l},\ G_{\gamma}^{(2)},\ G_{\delta}^{(2)}\}.$$

2.5 Повнота доведення

Тепер покажемо, що з такою перевіркою протокол GRO-16 дійсно має властивість повноти, тобто якщо P дійсно володіє монетою, то він завжди це доводить, тобто рівність (2.8) виконується. Доведення те ж саме, що й у попередньому підрозділі. Вважаємо, що доведення дійсне.

Розглянемо ліву частину рівності (2.8):

$$V_{AB} = e\left(\left(\alpha + \sum_{i=0}^{m} a_i u_i(x) + r\delta\right)G_1, \left(\beta + \sum_{i=0}^{m} a_i v_i(x) + s\delta\right)G_2\right) =$$
$$= e(G_1, G_2)^{(\alpha+\sum_{i=0}^{m} a_i u_i(x)+r\delta)(\beta+\sum_{i=0}^{m} a_i v_i(x)+s\delta)} = e(G_1, G_2)^{P_{left}},$$

де

$$P_{left} = \alpha\beta + \sum_{i=0}^{m} a_i \alpha v_i(x) + \alpha s\delta + \sum_{i=0}^{m} a_i \beta u_i(x) + \sum_{i=0}^{m} a_i u_i(x) \cdot \sum_{i=0}^{m} a_i v_i(x) +$$
$$+ s\delta \sum_{i=0}^{m} a_i u_i(x) + r\delta\beta + r\delta \sum_{i=0}^{m} a_i v_i(x) + rs\delta^2 =$$
$$= \alpha\beta + \sum_{i=0}^{m} a_i(\beta u_i(x) + \alpha v_u(x)) + s\delta\left(\alpha + \sum_{i=0}^{m} a_i u_i(x)\right) +$$
$$+ r\delta\left(\beta + \sum_{i=0}^{m} a_i v_i(x)\right) + rs\delta^2 + \sum_{i=0}^{m} a_i u_i(x) \cdot \sum_{i=0}^{m} a_i v_i(x).$$

Розглянемо тепер праву частину рівності (2.8):

$$V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i \cdot V_{\alpha\beta\gamma,i}\right)V_{C\delta} = e(G_1, G_2)^{\alpha\beta} e(G_1, G_2)^{\sum_{i=0}^{m} a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))} \times$$
$$\times e(G_1, G_2)^{h(x)t(x)+s\delta\left(\alpha+\sum_{i=0}^{m} a_i u_i(x)+r\delta\right)} e(G_1, G_2)^{r\delta\left(\beta+\sum_{i=0}^{m} a_i v_i(x)+s\delta\right)-rs\delta^2} =$$
$$= e(G_1, G_2)^{P_{right}},$$

де

$$P_{right} = \alpha\beta + \sum_{i=0}^{m} a_i(\beta u_i(x) + \alpha v_i(x)) + rs\delta^2 + s\delta\left(\alpha + \sum_{i=0}^{m} a_i u_i(x)\right) + rs\delta^2 +$$
$$+ r\delta\left(\beta + \sum_{i=0}^{m} a_i v_i(x)\right) - rs\delta^2 + \sum_{i=0}^{m} a_i w_i(x) + h(x)t(x).$$

Легко бачити, що якщо скоротити однакові доданки в обох частинах, то отримаємо $P_{left} = P_{right}$. Якщо доведення дійсне, то P володіє такими $\{a_i\}_{i=0}^{m}$, що рівність (2.4) виконується.

Висновки до розділу 2

У цьому розділі на прикладу протоколу GRO-16 було розглянуто принцип функціонування доведень без розголошення типу zk-SNARK. Зазначимо, що найбільш трудомістку фазу zk-SNARK становить фаза SETUP. Тому дуже актуальним питанням є питання оптимізації цієї фази, наприклад, коштом розпаралелювання його формування. Але робити це потрібно дуже обережно, тому що деякі оптимізації можуть призвести до втрати стійкості протоколу. У подальшому викладі буде як раз розглянуто питання його оптимізації та питання втрати чи збереження стійкості під час певних оптимізацій.

3 АТАКИ НА ПРОТОКОЛ GRO-16

У цьому розділі описані умови, за яких досить можливо виконати успішну підготовку до атак. Також, атаки строго формалізовані та доведено їх успіх при успішній підготовці до них.

3.1 Аналіз умов, необхідних для реалізації атак

Доцільно вважати, що знання хоча б одного елементу із критичних елементів налаштування є досить важкою задачею, як у математиці, так і використовуючи знання інших областей, зокрема, інформаційної безпеки.

Але розглянемо досить імовірний сценарій, при якому імовірність успішно провести атаку набагато вище, чим якби учасники блокчейну обрали довший, більш ресурсомісткий, але тим самим більш безпечний спосіб.

Нагадаємо та розглянемо декілька моментів, пов'язаних з етапом SETUP та самою множиною *Setup*.

Як було зазначено в 1.2.2, перевагою zk-SNARK'ів є «стислість» доведення, тобто вони швидко генеруються, мало важать (доведення для GRO-16 важить 288 бітів), швидко передаються і швидко перевіряються. Здавалося б, одні плюси, але така зручність пов'язана із досить довгим і громіздким етапом SETUP, у якому формується множина *Setup*, з якої вже можна отримати ключі PK і VK для побудови доведення та його перевірки відповідно. Етап SETUP може займати понад добу, а множина *Setup* займати понад сто гігабайтів. Зрозуміло, що передача такої великої кількості даних також означає доволі довгу передачу, при передачі якої можуть бути пошкодження, що призводить до ще більших витрат часу. Тим більше, усі учасники повинні попередньо пройти автентифікацію та потім бути онлайн, тому що хоча б без одного учасника сформувати

Setup не можна (не по протоколу). І також зрозуміло, що зовсім недоцільно створювати множину налаштування кожний раз, коли хтось з учасників хоче провести транзакцію. Тоді у кращому випадку буде виконуватись одна транзакція у дві доби, що не припустимо. Тому виходить, що етап SETUP треба просто перестраждати, щоб на досить довгий час користуватись множиною *Setup*.

Але учасники блокчейну можуть вирішити змінювати налаштування частіше, формувати його швидше та витрачати на порядок менше ресурсів, якщо оберуть наступний сценарій. Вони можуть розпаралелити формування налаштування між випадково обраними групами учасників так, що кожна група обчислює виданий їм елемент, а потім скласти множину *Setup*. Дійсно, такий спосіб обчислення налаштування має свої досить суттєві переваги. Але оскільки також цілком доцільно вважати, що у блокчейні значна частка учасників потенційно не мають наміру діяти чесно, повинні враховуватись ризики можливості, що якесь із критичних елементів налаштування буде генеруватися групою, що цілком складається із нечесних учасників.

Також це значно полегшує зловмисникам задачу підготовки до атак, тому що стає значно менша кількість учасників, яка формувала той чи інший критичний елемент налаштування. Так, від учасника може вимагатись, щоб він видаляв свій елемент розкладу одразу ж після його участі у формування налаштування. Але, якщо не існують допоміжні методи контролю, учасники можуть просто це не робити, тому що забули, не захотіли, не розуміють ризики, або будь-який інший людський фактор. І цей елемент розкладу може бути випадково оголошений, або зберігатися нехитрим чином на ЕОМ, до якої можуть отримати доступ зловмисники або фізично, що важче, або через мережу Інтернет, що набагато легше. Останнє більш імовірно враховуючи, що навряд чи звичайна людина має достатньо навичок та знань, щоб захиститись від можливих атак на свою ЕОМ з ціллю дізнатись її елемент розкладу. Саме це і малось на увазі, коли було написано про інші області ризиків.

Саме тому у цьому розділі розроблені атаки на протокол GRO-16 та сформульовані твердження щодо їх успіху. Як було зазначено раніше, атаки базуються на знанні деяких критичних елементів *Setup* (2.7). Зауважимо, що атаки базуються не на вразливостях GRO-16. Основою для цих атак є людський фактор та їх нечітке виконання усіх необхідних правил виконання протоколу.

3.2 Формалізація атак та доведення їх успішності

Перед тим, як розглядати атаки, потрібно пояснити деякі процесі, пов'язані з проведенням і прийняттям транзакцій. Кожну монету зі значеннями $\{a_i\}_{i=0}^{m}$, де $a_0 = 1$ можна витратити тільки один раз. Після того, як було побудовано дійсне доведення володіння цією монетою та проведена транзакція, після якої у монети з'явився новий власник, для неї генеруються нові значення $\{a_i^{'}\}_{i=0}^{m}$ таким чином, що неможливо передбачити її нові значення. Для протоколу це означає, що одразу ж після чесного доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$ ідентифікатор $\{a_i\}_{i=0}^{l}$ одразу ж стає недійсним.

Це означає, що повторно використовувати доведення не має сенсу.

Як також було зазначено в 2.2, ніхто у блокчейні не знає критичні елементи *Setup* $\alpha, \beta, \gamma, \delta, x^i \in \mathbb{F}_p, i = \overline{1,n}$. Це потрібно тому, що знання будь-якого з цих елементів дасть зловмиснику M можливість підроблювати доведення і красти монети інших учасників так, що ніхто про це не дізнається. Покажемо, що наступні твердження дійсні.

Атака 3.1. *Викрадення монети з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *використовуючи знання* α *та* β *із Setup* (2.7).

Вхід: Setup, α, β, $\{a_i\}_{i=0}^{l}$.

Вихід: дійсне доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$.

1) Випадково обрати значення $C,\ D \in \mathbb{F}_p$.

2) Обчислити наступні значення:

а) $B^{(2)} = D^{-1}G_2$;

б) $C^{(1)} = CG_1$;

в) $T = \alpha G_\beta^{(1)}$;

г) $T_{1,i} = (\beta u_i(x)G_1),\ i = \overline{0,m}$;

д) $T_{2,i} = (\alpha v_i(x)G_1),\ i = \overline{0,m}$;

е) $A^{(1)} = D(T + \sum_{i=0}^{l} a_i(T_{1,i} + T_{2,i} + w_i(x)G_1) + C^{(1)})$.

Коректність даної атаки доводиться таким твердженням.

Твердження 3.1. *Якщо учасник* M *знає значення* α *та* β *із Setup* (2.7) *і хоче викрасти монету з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *то атака 3.1 спрацює.*

Доведення. Дійсно, розглянемо четвірку $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$, яку отримаємо на виході алгоритму 3.1. Учасник V виконує наступні кроки.

1) V обчислює

$$V_{AB} = e(A^{(1)}, B^{(2)}) = \\ = e(G_1, G_2)^{T+\sum_{i=0}^{l} a_i(T_{1,i}+T_{2,i}+w_i(x))+C\delta} = \\ = e(G_1, G_2)^{\alpha\beta+\sum_{i=0}^{l} a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))+C\delta} = \\ = e(G_1, G_2)^{P_{left}},$$

де

$$P_{left} = \alpha\beta + \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + C\delta.$$

2) V обчислює

$$V_{\alpha\beta}\left(\sum_{i=0}^{m} a_i \cdot V_{\alpha\beta\gamma,i}\right)V_{C\delta} = \\ = e(G_1, G_2)^{\alpha\beta} e(G_1, G_2)^{\sum_{i=0}^{l} a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))} e(G_1, G_2)^{C\delta} = \\ = e(G_1, G_2)^{P_{right}},$$

де

$$P_{right} = \alpha\beta + \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + C\delta.$$

3) V перевіряє рівність

$$V_{AB} = V_{\alpha\beta}\Big(\sum_{i=0}^{l} a_i V_{\alpha\beta\gamma,i}\Big) V_{C\delta}.$$

Легко бачити, що $P_{left} = P_{right}$, тобто рівність попереднього пункту виконується та V приймає доведення як дійсне. □

Перед алгоритмом 2.3 йшла мова про те, що спочатку необхідно обчислювати, наприклад, $\frac{u_i(x)}{\gamma}G_1$, а вже потім $\frac{\beta u_i(x)}{\gamma}G_1$. Пояснювалось це тим, що витоки інформації у вигляді $\beta u_i(x)G_1$ та $\alpha v_i(x)G_1$ можуть значно спростити підготовку до атаки учаснику M, $i = \overline{0, m}$. Тепер покажемо це у вигляді наступних двох тверджень.

Атака 3.2. *Викрадення монети з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *використовуючи знання* α *із Setup* (2.7).

Вхід: Setup, α, $\{a_i\}_{i=0}^{l}$.

Вихід: дійсне доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$.

1) Випадково обрати значення $C,\ D \in \mathbb{F}_p$.

2) Обчислити наступні значення:

а) $(\alpha^{-1} \bmod p)$ *та* $(D^{-1} \bmod p)$;

б) $B^{(2)} = D^{-1}G_2$;

в) $C^{(1)} = CG_1$;

г) $T_1 = \alpha G_\beta^{(1)}$;

д) $T_2 = \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))G_1$;

е) $T_3 = CG_\delta^{(1)}$;

ж) $A^{(1)} = D(T_1 + T_2 + T_3)$.

Коректність даної атаки доводиться таким твердженням.

Твердження 3.2. *Нехай учасники спочатку обчислюють та публікують значення* $\beta u_i(x)G_1$ *та* $\alpha v_i(x)G_1$, $i = \overline{0, m}$. *Якщо учасник* M

знає значення α із Setup (2.7) і хоче викрасти монету з ідентифікатором $\{a_i\}_{i=0}^{l}$, то атака 3.2 спрацює.

Доведення. Дійсно, розглянемо четвірку $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$, яку отримаємо на виході алгоритму 3.2. Учасник V виконує наступні кроки.

1) V обчислює

$$V_{AB} = e(A^{(1)}, B^{(2)}) = \\ = e(G_1, G_2)^{T_1+T_2+T_3} = e(G_1, G_2)^{\alpha\beta+\sum_{i=0}^{l} a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))+C\delta} = \\ = e(G_1, G_2)^{P_{left}},$$

де

$$P_{left} = \alpha\beta + \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + C\delta.$$

2) V обчислює

$$V_{\alpha\beta}\left(\sum_{i=0}^{m} a_i \cdot V_{\alpha\beta\gamma,i}\right) V_{C\delta} = \\ = e(G_1, G_2)^{\alpha\beta} e(G_1, G_2)^{\sum_{i=0}^{l} a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))} e(G_1, G_2)^{C\delta} = \\ = e(G_1, G_2)^{P_{right}},$$

де

$$P_{right} = \alpha\beta + \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + C\delta.$$

3) V перевіряє рівність

$$V_{AB} = V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i V_{\alpha\beta\gamma,i}\right) V_{C\delta}.$$

Легко бачити, що $P_{left} = P_{right}$, тобто рівність попереднього пункту виконується та V приймає доведення як дійсне. □

Аналогічна атака стосується знання β.

Атака 3.3. *Викрадення монети з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *використовуючи знання* α *із Setup* (2.7).

Вхід: Setup, α, $\{a_i\}_{i=0}^{l}$.

Вихід: дійсне доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$.

1) Випадково обрати значення $C,\ D \in \mathbb{F}_p$.

2) Обчислити наступні значення:

а) $(\alpha^{-1} \bmod p)$ *та* $(D^{-1} \bmod p)$;

б) $B^{(2)} = D^{-1}G_2$;

в) $C^{(1)} = CG_1$;

г) $T_1 = \beta G_\alpha^{(1)}$;

д) $T_2 = \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))G_1$;

е) $T_3 = CG_\delta^{(1)}$;

ж) $A^{(1)} = D(T_1 + T_2 + T_3)$.

Коректність доводиться аналогічним твердженням.

Твердження 3.3. *Нехай учасники спочатку обчислюють та публікують значення* $\beta u_i(x)G_1$ *та* $\alpha v_i(x)G_1$, $i = \overline{0, m}$. *Якщо учасник* M *знає значення* β *із Setup* (2.7) *і хоче викрасти монету з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *то атака 3.3 спрацює.*

Доведення. Доведення абсолютно аналогічне доведенню твердження 3.2. □

Тобто, напрошується висновок, що проста зміна послідовності множення елементів може значно спростити підготовку до атаки учаснику M. І хоча спочатку це може здатися не чимось критичним. Однак це визначає, достатньо зловмиснику знати лише або α, або β; або йому потрібно знати їх одночасно, або знати якийсь з них та γ. Зрозуміло, що складність цих цілей значно відрізняється.

Знання інших критичних елементів налаштування не потребує додаткових знань.

Атака 3.4. *Викрадення монети з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *використовуючи знання* δ *із Setup* (2.7).

Вхід: Setup, δ, $\{a_i\}_{i=0}^{l}$.

Вихід: дійсне доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$.

1) Випадково обрати значення $D \in \mathbb{F}_p$.

2) Обчислити наступні значення:

а) $(\delta^{-1} \bmod p)$ *та* $(D^{-1} \bmod p)$;

б) $A^{(1)} = DAG_1$, *де*

$$A = \alpha + \sum_{i=0}^{m} a_i u_i(x) + r\delta;$$

в) $B^{(2)} = D^{-1}BG_2$, *де*

$$B = \beta + \sum_{i=0}^{m} a_i v_i(x) + s\delta;$$

г)

$$T_1 = \sum_{i=0}^{l} \frac{a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} G_1;$$

д)

$$T_2 = \sum_{i=0}^{l} \frac{a_i^{'}(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} G_1;$$

е) $C^{(1)} = CG_1 + T_1 - T_2$, *де*

$$C = \frac{\sum_{i=l+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} + \frac{h(x)t(x)}{\delta} + sA + rB + rs\delta.$$

Коректність даної атаки доводиться таким твердженням.

Твердження 3.4. *Якщо учасник* M *знає значення* δ *із Setup* (2.7) *і хоче викрасти монету з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *то атака 3.3 спрацює.*

Доведення. Дійсно, розглянемо четвірку $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$, яку отримаємо на виході алгоритму 3.4. Учасник V виконує наступні кроки.

1) V обчислює

$$V_{AB} = e(G_1, G_2)^{P_{left}},\ \text{де}$$

$$P_{left} = \alpha\beta + \sum_{i=0}^{m} a_i(\beta u_i(x) + \alpha v_u(x)) + s\delta\left(\alpha + \sum_{i=0}^{m} a_i u_i(x)\right) + r\delta\left(\beta + \sum_{i=0}^{m} a_i v_i(x)\right) + rs\delta^2 + \sum_{i=0}^{m} a_i u_i(x) \cdot \sum_{i=0}^{m} a_i v_i(x).$$

2) V обчислює

$$V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i^{'} V_{\alpha\beta\gamma,i}\right) V_{C\delta} =$$

$$= e(G_1, G_2)^{\alpha\beta} \cdot e(G_1, G_2)^{\sum_{i=0}^{l} a_i^{'} \cdot (\beta \cdot u_i(x) + \alpha \cdot v_i(x) + w_i(x))} \times$$

$$\times e(G_1, G_2)^{\sum_{i=l+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + h(x)t(x) + sA\delta + rB\delta + rs\delta^2} \times$$

$$\times e(G_1, G_2)^{\sum_{i=0}^{l} a_i(\beta \cdot u_i(x) + \alpha v_i(x) + w_i(x))} e(G_1, G_2)^{-\sum_{i=0}^{l} a_i^{'} \cdot (\beta u_i(x) + \alpha v_i(x) + w_i(x))},\ \text{де}$$

$$P_{right} = \alpha\beta + \sum_{i=0}^{m} a_i(\beta u_i(x) + \alpha v_i(x)) + rs\delta^2 + s\delta\left(\alpha + \sum_{i=0}^{m} a_i u_i(x)\right) + rs\delta^2 + r\delta\left(\beta + \sum_{i=0}^{m} a_i v_i(x)\right) - rs\delta^2 + \sum_{i=0}^{m} a_i w_i(x) + h(x)t(x).$$

3) V перевіряє рівність

$$V_{AB} = V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i V_{\alpha\beta\gamma,i}\right) V_{C\delta}.$$

Легко бачити, що $P_{left} = P_{right}$, тобто рівність попереднього пункту виконується та V приймає доведення як дійсне. □

Зауваження. Взагалі кажучи, учаснику M не потрібно володіти монетами. Він може використати чуже доведення

$\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$ і просто замінити $C^{(1)}$ на $C_M^{(1)}$, де

$$C_M^{(1)} = C^{(1)} + \sum_{i=0}^{l} \frac{a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} - \\ - \sum_{i=0}^{l} \frac{a_i^{'}(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta}.$$

Зауваження. Зокрема, з твердження зрозуміло, що коли ми генеруємо налаштування (2.7), значення γ не повинно дорівнювати δ. В іншому випадку не потрібно навіть знати значення δ, щоб генерувати подібні доведення.

Остання атака.

Атака 3.5. *Викрадення монети з ідентифікатором* $\{a_i\}_{i=0}^{l}$, *використовуючи знання* x *із Setup* (2.7).

Вхід: Setup, x, $\{a_i\}_{i=0}^{l}$.

Вихід: дійсне доведення $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$.

1) Випадково обрати значення $D \in \mathbb{F}_p$.

2) Обчислити $t(x)$ *із 2.4. Зауважимо, що можна це зробити, тому що поліном* $t(x)$, *тобто іншими словами — його коефіцієнти, відомі всім учасникам. Надалі* $t(x) \equiv t$.

3) Обчислити $(t^{-1} \bmod p)$ *та* $(x^{-1} \bmod p)$.

4) Нагадаємо, що із налаштування (2.7) *усім учасникам відомо значення*

$$T_{\delta,1}^{(1)} = \frac{xt(x)}{\delta} G_1.$$

Помножити $T_{\delta,1}^{(1)}$ *на* $t^{-1}x^{-1}$ *і отримаємо* $\delta^{-1}G_1$.

5) Обчислити наступні значення:

а) $T_i = a_i w_i(x),\ i = \overline{0,\ l}$;

б)

$$W^{(1)}(x) = \sum_{i=0}^{l} \frac{T_i}{\delta} G_1.$$

в) $T_{i,j} = a_i u_i(x) a_j v_j(x),\ i = \overline{0,\ l},\ j = \overline{0,\ l}.$

г)

$$D^{(1)}(x) = \sum_{i=0}^{l} \sum_{j=0}^{l} \frac{T_{i,j}}{\delta} G_1.$$

д) $A^{(1)} = AG_1$, *де*

$$A = \alpha + \sum_{i=0}^{l} a_i u_i(x).$$

е) $B^{(2)} = BG_2$, *де*

$$B = \beta + \sum_{i=0}^{l} a_i v_i(x).$$

ж) $C^{(1)} = D^{(1)}(x) - W^{(1)}(x)$.

Коректність даної атаки доводиться таким твердженням.

Твердження 3.5. *Якщо учасник M знає значення x із Setup* (2.7) *і хоче викрасти монету з ідентифікатором $\{a_i\}_{i=0}^{l}$, то атака 3.5 спрацює.*

Доведення. Дійсно, розглянемо четвірку $\langle A^{(1)},\ B^{(2)},\ C^{(1)},\ \{a_i\}_{i=0}^{l}\ \rangle$, яку отримаємо на виході алгоритму 3.5. Учасник V виконує наступні кроки.

1) V обчислює

$$V_{AB} = e(A^{(1)}, B^{(2)}) = \\ = e(G_1, G_2)^{(\alpha + \sum_{i=0}^{l} a_i u_i(x))(\beta + \sum_{i=0}^{l} a_i v_i(x))} = \\ = e(G_1, G_2)^{P_{left}},$$

де

$$P_{left} = \alpha\beta + \sum_{i=0}^{l} a_i \alpha v_i(x) + \sum_{i=0}^{l} a_i \beta u_i(x) + \sum_{i=0}^{l} \sum_{j=0}^{l} a_i u_i(x) a_j v_j(x).$$

2) V обчислює

$$V_{\alpha\beta}\left(\sum_{i=0}^{m} a_i \cdot V_{\alpha\beta\gamma,i}\right) V_{C\delta} =$$
$$= e(G_1, G_2)^{\alpha\beta} e(G_1, G_2)^{\sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))} \times$$
$$\times e(G_1, G_2)^{\sum_{i=0}^{l}\sum_{j=0}^{l} a_i u_i(x) a_j v_j(x) - \sum_{i=0}^{l} a_i w_i(x)} = e(G_1, G_2)^{P_{right}},$$

де

$$P_{right} = \alpha\beta + \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) +$$
$$+ \sum_{i=0}^{l}\sum_{j=0}^{l} a_i u_i(x) a_j v_j(x) - \sum_{i=0}^{l} a_i w_i(x).$$

3) V перевіряє рівність

$$V_{AB} = V_{\alpha\beta}\left(\sum_{i=0}^{l} a_i V_{\alpha\beta\gamma,i}\right) V_{C\delta}.$$

Легко бачити, що $P_{left} = P_{right}$, тобто рівність попереднього пункту виконується та V приймає доведення як дійсне. □

Зауваження. Загалом, учаснику M достатньо знати будь-яку степінь x. Є методи, що дозволяють теоретично знайти x за поліноміальний час. Тому доцільно вважати, що знання про будь-яку степінь x теж може порушити вагомість протоколу.

3.3 Стратегія захисту від приведених атак шляхом використання пристрою TPM

Оскільки секретність значень критичних елементів налаштування ставить дуже гостре питання щодо вагомості zk-SNARK протоколу GRO-16, треба приділити дуже велику увагу до секретності їх елементів

розкладу.

Очевидно, що чим довше елементи розкладу зберігаються на ЕОМ, тим більше очікувано більша імовірність знання одного з критичних елементів налаштування. Як було зазначено раніше, це пов'язано з двома причинами.

1) Це нечесний учасник, який може змовитись з іншими учасниками, щоб дізнатись якесь значення із критичних елементів налаштування.

2) Це можливість для учасника отримати доступ до пам'яті ЕОМ іншого учасника, до якої можуть отримати доступ зловмисники або фізично, що важче, або через мережу Інтернет, що на порядок легше, та дізнатись його секретний елемент розкладу.

Тобто, у будь-якому випадку бажано видаляти елементи розкладу одразу ж, як у них немає потреби для протоколу

Одним із рішень є вимога до кожного учасника наявності ДПМ, який буде:

– генерувати елементи розкладу;

– зберігати їх у зашифрованому вигляді так, щоб навіть власник ДПМ їх не знав;

– виконувати операції множення у необхідній групі;

– безпечно видаляти елементи розкладу, коли відповідні елементи налаштування будуть згенеровані.

Тут не буде розглянута повна робота таких пристроїв, як, наприклад, шифрування, розшифрування, перевірка правильності шифрування та видалення інформації. Надалі розглянуті лише основні поняття та відповіді на можливі запитання, а також додатковий алгоритм обчислення і імплементація. Детальний опис пристроїв наведено у [15]; далі викладаються основні положення, необхідні для пояснення механізмів роботи гарантованого видалення елементів розкладу кожного учасника одразу ж після його участі у формуванні елементів *Setup*.

Оригінальна стаття є важливим прикладом того, що такі пристрої взагалі існують, і що можливо замовити подібний пристрій з необхідними

функціями.

Отже, нехай учасники блокчейну замовляють ДПМ як модифікацію на основі існуючого ДПМ у компанії-розробника. Відповідно до [15], вважаємо, що компанія-розробник не має інтересів навмисно встановлювати шкідливе програмне забезпечення у свої пристрої, окрім непередбачених програмних помилок та коду, який примусила додати державна охоронна установа. Іншими словами, вважається, що не плануються атаки на конкретного користувача.

Кожен ДПМ на стадії виробництва ініціалізації «на льоту» генерує унікальну пару ключів АЦПЕК: Prv_t і Pub_t. Другий ключ публікується на сайті виробника та відомий усім. Перший ключ безпечно зберігається у ДПМ і його не знає ніхто. Таким чином кожний інший учасник V може перевірити наявність ДПМ в учасника P:

1) P створює повідомлення.

2) Використовує відповідну програму, щоб створити запит до ДПМ на підпис повідомлення.

3) Публікує трійку ⟨Повідомлення, ЦП, свій публічний ключ⟩.

4) V шукає на сайті виробника публічний ключ, наданий P.

5) Якщо V його знаходить, перевіряє підпис. В іншому випадку P намагається обманути.

6) Якщо підпис дійсний, V переконується, що P має ДПМ. В іншому випадку P намагається обманути.

Отже, розглянемо декілька алгоритмів генерації елементів налаштування (2.7). У кінці буде наведено повний алгоритм формування налаштування.

Першим чином, як буде видно далі, учасникам необхідно обчислити

$$\{X_i^{(1)}\}_{i=0}^{s+n-1}, X_i^{(1)} = x^i G_1, \tag{3.1}$$

$$\{X_i^{(2)}\}_{i=0}^{s+n-1}, X_i^{(2)} = x^i G_2, \tag{3.2}$$

де s — степінь полінома $t(y)$. Можна помітити, що тут обчислюються більші

степені x, ніж приведено в (2.7). Це тому, що кожний елемент множини

$$\{x^i t(x) G_1\}_{i=0}^{n-2},$$

буде обчислюватись як

$$x^i t(x) G_1 = t_s x^{s+i} + t_{s-1} x^{s+i-1} + \cdots + t_0 x^i G_1,\ i = \overline{0, n-2},$$

де t_j — коефіцієнти полінома $t(y)$, $j = \overline{0, s}$, який відомий всім учасникам, відповідно до 2.1. Таке обчислення пов'язано з тим, що після створення запиту до ДПМ на обчислення i–ї частини x, ДПМ обчислював усі необхідні для налаштування значення, та одразу ж видаляв відповідний x_i. Таким чином планується, що загалом елемент розкладу критичного елементу налаштування x_i буде існувати у пам'яті комп'ютера декілька секунд, що значно знижує імовірність дізнатись це значення. Такі міркування стосуються також усіх критичних елементів налаштування та γ. Доцільним буде вважати, що тільки дуже малий відсоток учасників зможуть виконати методи атаки на ДПМ, зокрема дуже низькорівневий реверс-інжинірінг. Також доцільним буде вважати, що кількість витрачених ресурсів на здобуття значення елементу розкладу буде значно вищою, ніж це значення може потенціально принести, адже само по собі воно не має ніякої цінності.

Розглянемо алгоритм взаємодії учасника P_i з його ДПМ при його участі в обчисленні $x^j G_1,\ x^j G_2,\ i = \overline{1, s+n-1}$. Нехай попередні P_q, $1 \leq q \leq i-1$, учасники опублікували четвірки значень $\langle \{U^j_{1,2,\ldots,q}\},\ \{T^j_{1,2,\ldots,q}\},\ \{U^j_q\},\ \{T^j_q\} \rangle,\ j = \overline{1, s+n-1}$, де

$$U^j_{1,2,\ldots,q} = x^j_q x^j_{q-1} \ldots x^j_2 x^j_1 G_1,$$
$$T^j_{1,2,\ldots,q} = x^j_q x^j_{q-1} \ldots x^j_2 x^j_1 G_2,$$
$$U^j_q = x^j_q G_1,\ T_j = x^j_q G_2,$$

використовуючи той самий алгоритм, приведений нижче, який наразі буде

використовувати P_i для внесення свого вкладу.

Алгоритм 3.1. Обчислення учасником P_i четвірки

$$\langle \{U^j_{1,2,\dots,i}\},\ \{T^j_{1,2,\dots,i}\},\ \{U^j_i\},\ \{T^j_i\}\rangle,\ j = \overline{1, s+n-1}$$

за допомогою ДПМ. Вхід: четвірка попереднього учасника

$$\langle \{U^j_{1,2,\dots,i-1}\},\ \{T^j_{1,2,\dots,i-1}\},\ \{U^j_{i-1}\},\ \{T^j_{i-1}\}\rangle,\ j = \overline{1, s+n-1}$$

.

1) Учасник P_i створює відповідний запит до ДПМ щодо обчислення своєї четвірки, подаючи у якості входу четвірку попереднього учасника.

2) ДПМ виконує на ступні кроки.

а) ДПМ випадково генерує значення $x_i \in \mathbb{F}_p$.

б) ДПМ обчислює множину значень $\{x^j_i\},\ j = \overline{1, s+n-1}$.

в) ДПМ обчислює значення $U^j_{1,2,\dots,i} = x^j_i U^j_{1,2,\dots,i-1},\ j = \overline{1, s+n-1}$.

г) ДПМ обчислює значення $T^j_{1,2,\dots,i} = x^j_i T^j_{1,2,\dots,i-1},\ j = \overline{1, s+n-1}$.

д) ДПМ обчислює значення $U^j_i = x^j_i U^j_{i-1},\ j = \overline{1, s+n-1}$.

е) ДПМ обчислює значення $T^j_i = x^j_i T^j_{i-1},\ j = \overline{1, s+n-1}$.

ж) ДПМ використовує функцію *Encrypt*, щоб зашифрувати множину значень $\{x^j_i\},\ j = \overline{1, s+n-1}$.

з) ДПМ використовує функцію *Audit*, щоб перевірити правильність шифрування множини значень $\{x^j_i\},\ j = \overline{1, s+n-1}$.

и) ДПМ використовує функцію *Delete*, щоб видалити секретний ключ, на якому була зашифрована множина значень $\{x^j_i\},\ j = \overline{1, s+n-1}$.

к) ДПМ генерує та створює цифровий підпис відсутність помилок при видаленні S_1 своїм секретним ключем Prv_t.

л) ДПМ підписує повідомлення:

$$S_2 = Sig\Big(H\Big(\{U^j_{1,2,\dots,i}\}||\{T^j_{1,2,\dots,i}\}||\{U^j_i\}||\{T^j_i\}||S_1\Big)\Big)$$

своїм секретним ключем Prv_t.

м) ДПМ повертає підписи $S_1,\ S_2$ і четвірку

$$\langle\{U^j_{1,2,\dots,i}\},\ \{T^j_{1,2,\dots,i}\},\ \{U^j_i\},\ \{T^j_i\}\rangle,\ j=\overline{1,s+n-1}$$

.

3) P_i публікує четвірку та підписи, що повернув ДПМ.

4) Кожний учасник V перевіряє підпис S, використовуючи раніше прив'язаний до P_i публічний ключ його ДПМ.

5) Якщо підпис дійсний, учасник V перевіряє рівності:

$$e(U^j_i, G_2) = e(G_1, T^j_i),$$
$$e(U^j_{1,2,\dots,i}, G_2) = e(U^j_{1,2,\dots,i-1}, T^j_i),$$
$$e(G_1, T^j_{1,2,\dots,i}) = e(U^j_i, T^j_{1,2,\dots,i-1}),\ j=\overline{1,s+n-1}.$$

Якщо ж підпис не дійсний або хоча б одна з рівностей не виконується, учасник P намагається обманути.

Висновки до розділу 3

У цьому розділі було детально проаналізовано стійкість протоколу zk-SNARK та його різних оптимізованих модифікацій.

Були винайдені п'ять видів атак на zk-SNARK протокол GRO-16, які порушують властивість вагомості. Це означає, що зловмисник має змогу довести знання секрету, якого насправді не знає. Своєю чергою, для протоколу GRO-16, це дає йому змогу красти монети інших учасників необмежену кількість разів, доки не зміниться Setup, якщо він успішно виконає підготовку до будь-якого виду атаки. Успішна реалізація наведених видів атак можлива лише через знання критичних елементів налаштування, тому їх секретності потрібно приділити максимальну увагу.

Таким чином, найбільшу частину всього часу займає підготовка,

якщо зловмисник хоче реалізувати один із наведених видів атак на zk-SNARK протокол GRO-16. Дійсно, дізнатись хоча б один із критичних елементів налаштування являє собою доволі складну математичну задачу, яка на цей час не має поліноміального розв'язку. Проте потенційна вигода, яку може отримати зловмисник може коштувати всіх його зусиль та витрат.

Були запропоновані стратегії захисту від наведених у цій роботі атак. Деякі з них базуються на зміні алгоритму формування налаштування. Дотримання приведених алгоритмів значно ускладнить підготовку до деяких з наведених атак для зловмисника. Інші стратегії пов'язані із використанням пристроїв третьої сторони. Була розглянута робота подібного пристрою та розроблений алгоритм взаємодії з ним. Використання подібних пристроїв унеможливить знання критичних елементів налаштування.

ВИСНОВКИ

Результати даної роботи можна окреслити наступними пунктами.

1) Розроблено алгоритми формування як простих, так і більш складних елементів множини налаштування протоколу GRO-16 із застосуванням білінійних відображень. Перевагою даних алгоритмів є те, що вони змушують учасників діяти чесно протягом алгоритму. Особливістю запропонованого алгоритму формування більш складних елементів є те, що лише зміна порядку обчислень може значно ускладнити підготовку до атак для зловмисника.

2) Створено алгоритм перевірки доведення на основі білінійних відображень та доведено повноту цієї перевірки. Дана перевірка не залежить від кількості інформації, яка доводиться. Для перевірки потрібно лише два рази обчислити білінійне відображення, яке обчислюється за поліноміальний час.

3) Проаналізовано сценарій, в якому учасники намагаються зменшити час та кількість витрачених ресурсів на етапі SETUP шляхом розпаралелюванням формування множини налаштування. Було з'ясовано, що в результаті цього зловмисникам стає набагато простіше формувати критичні елементи налаштування, що дозволяє їм красти чужі монети в необмеженій кількості до поки буде дійсна множина налаштування.

4) Створено п'ять атак на протокол GRO-16, що використовують знання різних критичних елементів налаштування. Хоча запропоновані атаки потребують довгої підготовки, тобто пошук певного критичного елементу налаштування, успішна підготовка гарантує успішність відповідної атаки та дозволить зловмиснику, як було зазначено у попередньому пункті, присвоювати чужі фінанси у необмеженій кількості та при цьому не буди спійманим. Також, складність атак є такою ж, як і складність створення доведення.

5) Для формування елементів множини налаштування було запропоновано використання TPM пристроїв, які обирають елемент розкладу і виконують з ним операції таким чином, що навіть власник цього пристрою не знає свій елемент розкладу. Також, цей пристрій видаляє елемент розкладу та підписує повідомлення про успішність операції видалення своїм секретним ключем. Використання таких пристроїв робить майже неможливим підготовку до атак шляхом викрадення елементів розкладу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Bitcoin: A Peer-to-Peer Electronic Cash System [Електронний ресурс] / Satoshi Nakamoto – Режим доступу: https://bitcoin.org/bitcoin.pdf.

2. Кошелек для криптовалюты — как его создать и какой лучше: холодный, мультивалютный, аппаратный или онлайн-криптокошелек [Електронний ресурс] – 2020. – Режим доступу: https://bit.ly/30gbdU4.

3. Только тихо! Как работают анонимные криптовалюты [Електронний ресурс] – 2020. – Режим доступу: https://www.rbc.ru/crypto/news/5d0b544c9a794722cc4524e3.

4. Биткоин-миксеры топ: обзор лучших миксеров для перемешивания криптовалюты Bitcoin. Принцип работы, список сайтов, нюансы, преимущества и недостатки [Електронний ресурс] / Редакция Profinvestment.com – 2020. – Режим доступу: http://profinvestment.com/bitcoin-mixer/.

5. Анонимные платежи: Dash или Bitcoin+Миксеры? [Електронний ресурс] 2015. – Режим доступу: https://bitnovosti.com/2015/05/29/dash-iti-bitcoin-mixery/.

6. Как работают криптовалютные миксеры и анонимные кошельки [Електронний ресурс] – Режим доступу: https://decenter.org/ru/kak-rabotayut-kriptovalyutnye-miksery-i-anonimnye-koshelki.

7. A simple guide to safely and effectively tumbling (mixing) bitcoins [Електронний ресурс] – 2015. – Режим доступу: https://darknetmarkets.org/a-simple-guide-to-safely-and-effectively-mixing-bitcoins/.

8. Z-Pay [Електронний ресурс] – Режим доступу: https://z-pay.io/.

9. BATMTwo [Електронний ресурс] – Режим доступу: https://www.generalbytes.com/en/products/batmtwo.

10. Биржи криптовалют без верификации. Как вывести криптовалюту без верификации личности [Електронний ресурс] – 2020. – Режим доступу: https://privatefinance.biz/top-7-kriptobirzh-bez-verifikaczii/

#h2_12.

11. Что такое zk-SNARK? [Електронний ресурс] – Режим доступу: https://z.cash/ru/technology/zksnarks/.

12. Протоколы zk-SNARKs и zk-STARK: различия и возможности [Електронний ресурс] / Алексей Глуховский – 2019. – Режим доступу: https://www.youtube.com/watch?v=v7YQJ9ak4x4.

13. Аналіз загроз при повторному використанні налаштування у snark-доведенні за протоколом GRO-16 [Електронний ресурс] / Бещук Андрій – 2020. – Режим доступу: https://drive.google.com/file/d/1WYvUZCFoT9YuhyBvd4Ke-Q8X4ZjJwV4G/view.

14. On the Size of Pairing-based Non-interactive Argument [Електронний ресурс] / Jens Groth – 2016. – Режим доступу: https://ia.cr/2016/260.

15. Deleting Secret Data with Public Verifiability [Електронний ресурс] / Feng Hao and Dylan Clarke and Avelino Francisco Zorzo – 2015. – Режим доступу: https://ia.cr/2014/364.